Uniwersytet Ekonomiczny w Katowicach

Text mining. Klasyfikacja

Bogna Zacny

zima 2019/2020

Wydział Informatyki i Komunikacji
Katedra Inżynierii Wiedzy

Agenda

Wprowadzenie

Klasyfikacja

Znajdowanie sposobu odwzorowywania danych w zbiór **predefiniowanych** klas.

Klasyfikacja

Znajdowanie sposobu odwzorowywania danych w zbiór **predefiniowanych** klas.

Głównym celem klasyfikacji jest zbudowanie formalnego modelu zwanego klasyfikatorem. Na wejściu mamy zbiór treningowy przykładów, będących listą wartości atrybutów opisowych i wybranego atrybutu decyzyjnego. Wynikiem procesu klasyfikacji jest otrzymany model (klasyfikator), który przydziela każdemu przykładowi wartość atrybutu decyzyjnego w oparciu o wartości pozostałych atrybutów.

Klasyfikacja

Baza danych zawiera obiekty opisane atrybutami (cechami nazywanymi deskryptorami), z których:

- przynajmniej jeden jest atrybutem decyzyjnym,
- pozostałe to predyktory.

Wartości atrybutu decyzyjnego dzielą zbiór krotek na predefiniowane klasy, składające się z krotek o tej samej wartości atrybutu decyzyjnego.

Przykład

hello	busines	regard	replica	Spam
1	0	1	1	spam
1	0	1	0	ham
1	0	1	0	ham
0	1	1	0	spam
0	1	1	0	ham
0	1	1	1	spam
1	1	0	1	spam
1	0	0	1	ham

Ocena klasyfikatora

Procedura klasyfikacji

Budowa modelu składa się z dwóch faz:

- treningu (uczenia),
- testowania.

Baza danych dzielona jest na dwie części (najczęściej w proporcji 7:3) – zbiór treningowy i zbiór testowy.

Przykład

Spam	Predykcja
spam	ham
ham	ham
ham	ham
spam	spam
ham	spam
spam	spam
spam	spam
ham	ham

```
table(wynikPred$Predykcja,
      wynikPred$Spam)
```

```
##
##          ham spam
##    ham     3    1
##    spam    1    3
```

Ocena jakości klasyfikacji

Ocenę przydatności klasyfikatorów dokonuje się poprzez estymację błędu lub trafności klasyfikowania w odniesieniu do zbiorów testowych, dla których przynależność do klas poszczególnych przykładów jest znana.

W niektórych problemach istotne jest rozróżnienie błędów nieprawidłowego zakwalifikowania przykładu do innej klasy niż znana wartość atrybutu decyzyjnego (w medycynie zakwalifikowanie chorego pacjenta do zdrowych jest bardziej niebezpieczne niż odwrotna sytuacja).
W takich przypadkach do oceny zdolności klasyfikacyjnych badanych modeli przyjmuje się miary oparte na: **macierzy pomyłek** (*confusion matrix*).

Ocena jakości klasyfikacji - macierz pomyłek

Macierz pomyłek jest macierzą kwadratową o wymiarach $k \times k$, gdzie k stanowi liczbę klas decyzyjnych.

Wiersze macierzy zawierają informacje o liczbie obiektów przyporządkowanych wg predykcji modelu, natomiast w kolumnach umiejscowione są liczby rzeczywistych przyporządkowaniach przykładów do klas. Na przecięciu *i*-tego wiersza oraz *j*-tej kolumny umieszczana jest liczba przykładów zaliczonych przez klasyfikator do klasy *j*-tej a należącej do *i*-tej klasy.

Ocena jakości klasyfikacji - miary

Wrażliwość klasyfikatora to jego zdolność do wykrywania przypadków prawdziwie pozytywnych.
Specyficzność określa zdolność do wykrywania przypadków prawdziwie negatywnych.
Trafność wyraża stosunek liczby poprawnie zakwalifikowanych obserwacji do liczby wszystkich obserwacji.
Błąd klasyfikacji określa stosunek liczby niepoprawnie zakwalifikowanych obserwacji do liczby wszystkich obserwacji.

Przykład

```
confusionMatrix(wynikPred$Predykcja, wynikPred$Spam)
```

```
Confusion Matrix and Statistics

          Reference
Prediction ham spam
      ham    3    1
      spam   1    3

               Accuracy : 0.75
                  ...
             Sensitivity : 0.750
             Specificity : 0.750
                  ...
        'Positive' Class : ham
```

Metody klasyfikacji

Rodzaje algorytmów

- tabela częstości,
  - ZeroR
  - OneR
  - naiwny klasyfikator bayesowski
  - drzewa decyzyjne
- macierz kowariancji,
  - liniowa analiza dyskryminacyjna
  - regresja logistyczna
- funkcja podobieństwa,
  - k najbliższych sąsiadów
- inne
  - maszyna wektorów nośnych
  - klasyfikatory liniowe,
  - sieci neuronowe.

Klasyfikacja tekstu

Korpus

tekst	klasa
I had a peanut butter sandwich for breakfast.	food
I like to eat almonds, peanuts and walnuts.	food
My neighbor got a little dog yesterday.	animal
Cats and dogs are mortal enemies.	animal
You mustn't feed peanuts to your dog and cat.	animal
I ate peanuts on a walk with my dog.	food

Przygotowanie zbioru uczącego i testowego

```
set.seed(103)
zdania_wUcz <- createDataPartition(zdania_tab$klasa,
                                   p = 4/6, list = FALSE)
(zdania_train <- zdania_tab[zdania_wUcz,])
```

	tekst	klasa
1	I had a peanut butter sandwich for breakfast.	food
2	I like to eat almonds, peanuts and walnuts.	food
4	Cats and dogs are mortal enemies.	animal
5	You mustn't feed peanuts to your dog and cat.	animal

```
(zdania_test <- zdania_tab[-zdania_wUcz,])
```

	tekst	klasa
3	My neighbor got a little dog yesterday.	animal
6	I ate peanuts on a walk with my dog.	food

Przygotowanie macierzy dokument-term ze zmienną `klasa`

- Zbiór treningowy

```
df_zd_train <- data.frame(as.matrix(dtm_zd_train),
                          klasa = zdania_train$klasa)
```

almonds	breakfast	butter	cat	dog	eat	enemies	feed	like
0	1	1	0	0	0	0	0	0
1	0	0	0	0	1	0	0	1
0	0	0	1	1	0	1	0	0
0	0	0	1	1	0	0	1	0

mortal	mustn.t	peanut	sandwich	walnuts	klasa
0	0	1	1	0	food
0	0	1	0	1	food
1	0	0	0	0	animal
0	1	1	0	0	animal

Przygotowanie macierzy dokument-term ze zmienną `klasa`

- Zbiór testowy

```
(df_zd_test <- data.frame(as.matrix(dtm_zd_test),
                          klasa = zdania_test$klasa))
```

ate	dog	got	littl	neighbor	peanut	walk	yesterday	klasa
0	1	1	1	1	0	0	1	animal
1	1	0	0	0	1	1	0	food

Przygotowanie macierzy dokument-term ze zmienną `klasa`

Termy w zbiorze uczącym:

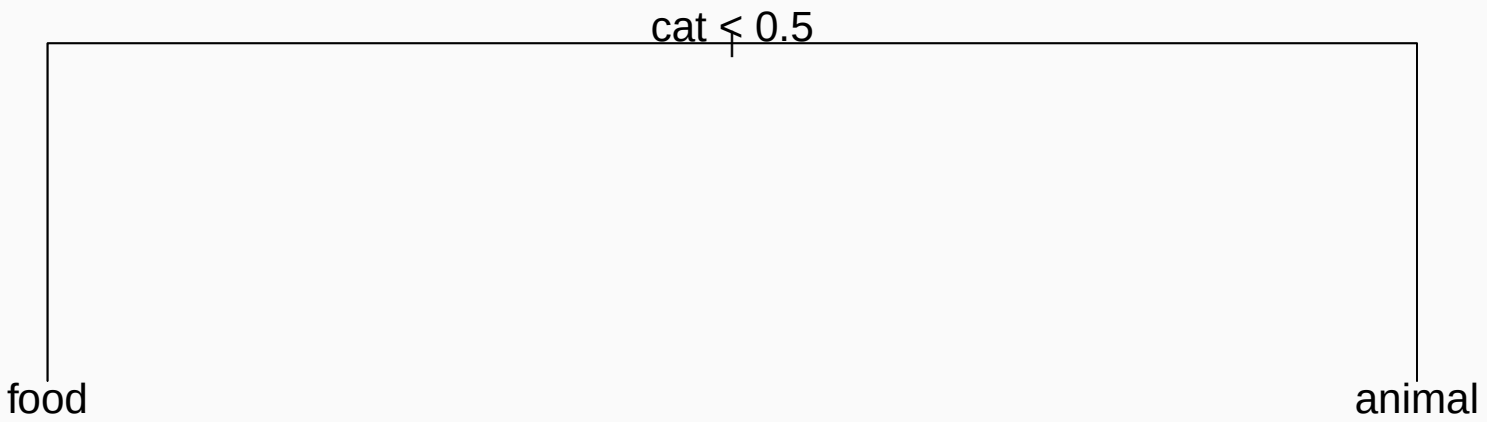
```
## [1] "almonds"   "breakfast" "butter"    "cat"       "dog"
## [1] "enemies" "feed"    "like"    "mortal"  "mustn.t"
## [1] "peanut"   "sandwich" "walnuts"  "klasa"
```

Termy w zbiorze testowym:

```
## [1] "ate"      "dog"      "got"      "littl"    "neighbor"
## [1] "peanut"    "walk"      "yesterday" "klasa"
```

Budowa drzewa decyzyjnego

```
model_zd_train <- tree(klasa ~ ., data = df_zd_train,
                       control = tree.control(6,
                                              mincut = 1,
                                              minsize = 2))
plot(model_zd_train);text(model_zd_train, pretty = 0, cex = 1.5)
```



Weryfikacja modelu

```
## node), split, n, deviance, yval, (yprob)
##       * denotes terminal node
##
## 1) root 4 5.545 animal ( 0.5 0.5 )
##   2) cat < 0.5 2 0.000 food ( 0.0 1.0 ) *
##   3) cat > 0.5 2 0.000 animal ( 1.0 0.0 ) *
```

```
df_zd_test
```

ate	dog	got	littl	neighbor	peanut	walk	yesterday	klasa
0	1	1	1	1	0	0	1	animal
1	1	0	0	0	1	1	0	food

Rozwiązanie

Na etapie tworzenia macierzy dokument term dla zbioru *testowego*, należy wskazać, że w macierzy tej mają pojawić się te same termy, które pojawiają się w macierzy dokument term dla zbioru *treningowego*

Stworzenie wektora termów

Pierwszym krokiem jest zapamiętanie, które termy wyodrębnione zostały dla zbioru treningowego.

```
(bow_zd <- findFreqTerms(dtm_zd_train, 0.95))
```

```
##  [1] "almonds"   "breakfast" "butter"    "cat"       "dog"
##  [6] "eat"       "enemies"   "feed"      "like"      "mortal"
## [11] "mustn't"   "peanut"    "sandwich"  "walnuts"
```

Stworzenie macierzy dokument term, dla zbioru testowego

Drugim, wykorzystanie "worka słów" jako słownika wykorzystanego do utworzenia macierzy dokument term dla zbioru testowego.

```
dtm_zd_test_bow <-  DocumentTermMatrix(korpus_zd_test,
                      control = list(stopwords = TRUE,
                                     stemming = TRUE,
                                     removePunctuation = TRUE,
                                     dictionary = bow_zd))
df_zd_test_bow <- data.frame(as.matrix(dtm_zd_test_bow),
                             klasa = zdania_test$klasa)
```

Stworzenie macierzy dokument term, dla zbioru testowego

almonds	breakfast	butter	cat	dog	eat	enemies	feed	like
0	0	0	0	1	0	0	0	0
0	0	0	0	1	0	0	0	0

mortal	mustn.t	peanut	sandwich	walnuts	klasa
0	0	0	0	0	animal
0	0	1	0	0	food

Weryfikacja modelu

```
przew_zd <- predict(model_zd_train, df_zd_test_bow,
                    type = "class")
confusionMatrix(przew_zd, df_zd_test$klasa)
```

```
Confusion Matrix and Statistics

          Reference
Prediction animal food
    animal      1    1
    food        0    0

               Accuracy : 0.5
                  ...
            Sensitivity : 1.0
            Specificity : 0.0
                  ...
       'Positive' Class : animal
```